Hybrid Biased k-NN to Predict Movie Tweets Popularity

Ladislav Peska
Faculty of Mathematics and Physics
Charles University in Prague
Malostranske namesti 25, Prague, Czech Republic
peska@ksi.mff.cuni.cz

Peter Vojtas
Faculty of Mathematics and Physics
Charles University in Prague
Malostranske namesti 25, Prague, Czech Republic
vojtas@ksi.mff.cuni.cz

ABSTRACT
In this paper we describe approach of our SemWexMFF group to the RecSys Challenge 2014: User Engagement as Evaluation. Target of the challenge was to predict level of user engagement on tweets generated automatically from IMDB. During experiments we have tested several state-of-the-art prediction techniques and proposed a variant of item based k-NN algorithm, which better reflects user engagement and nature of the movie domain content-based attributes. Our final solution (placed in the midfield of the challenge leaderboard) is an aggregation of several runs of this algorithm. In the paper we will further describe dataset used, data filtration, algorithm details and settings as well as decisions made during the challenge and dead ends we explored.

Categories and Subject Descriptors
H.3.3 [**Information Systems**]: Information Search and Retrieval - *Information Filtering*

General Terms
Measurement, Human Factors, Experimentation.

Keywords
Hybrid Biased k-NN, User Engagement, RecSys Challenge 2014, SemWexMFF team

1. INTRODUCTION
The task of 2014 RecSys Challenge was to predict user engagement on Twitter for movie rating tweets automatically posted from IMDB (from users, who connected their IMDB and Twitter accounts). The user engagement of each tweet was defined as a sum of retweets and favorites of this tweet. Other tweet data was also made available for use (including IMDB movie identifier), and participants were allowed to download additional metadata by themselves.

The test dataset contains large number of new movies unseen in the training data, so we expect that purely collaborative recommenders will not provide very good predictions. Another possible limitation is large number of zero user engagement causing problems to some classification based algorithms e.g. decision trees. The task is also not well suited for the purely content-based recommenders as there are new users in the test dataset and also many users posted only a few tweets.

The average user engagement in the train set is 0.216, over 95% of the tweets have zero user engagement and almost 80% of users received zero engagement for all of their tweets. The situation seems to be similar to the purchases on an e-commerce site, where our previous experiments [4] shown that hybrid algorithms highly outperformed collaborative ones. In [4] the Content-boosted Matrix Factorization (CBMF) [1] was used, however its time complexity has proved to be a problem during our experiments on the 2014 ESWC Challenge [3]. On the other hand the challenge winning method by Risotski et al. [5] showed that using relatively simple recommenders combined together may provide surprisingly good results.

2. OUR APPROACH
In our approach we worked with two main hypotheses:

1. Engagement of similar movies should be similar.
2. Engagement depends on neighborhood of the current user.

In order to define inter-movie similarity, we used IMDB querying API[1] to generate content-based attributes. We also considered using DBPedia or Freebase, but IMDB contains most of the relevant information and furthermore offers guaranteed 100% item coverage. Three types of attributes were downloaded: attributes describing popularity (*average rating*, *number of awards*, *IMDB metascore*), attributes related to widespread of the movie *(number of ratings*) and attributes about content (*movie name*, *release date*, *genre*, *country*, *language*, *director*, *actors*).

There is some room for improvement by using e.g. DBPedia *dcterms:subject*, ingoing / outgoing links or number of Wikipedia language editions, which we left for a future work.

The second hypothesis reflects our expectation that composition of user's friends and followers would greatly affect observed engagement. The twitter API contains only aggregated information (total numbers of friends and followers for each user), so we decided to use simple user bias instead of machine learning over user's friends.

2.1 Hybrid Biased k-NN
According to the hypothesis formulated in Section 2, we implemented a variant of well known item-based k-nearest-neighbors algorithm. Instead of e.g. collaborative similarity, the tweets are defined as similar, if the content-based similarity of their respective movies is high. The content-based similarity is an average of attributes similarities, which are defined according to their type. Similarity of *numeric* attributes (*average rating*, *number of ratings*, *number of awards*, *IMDB metascore* and *release date*) is defined as their difference normalized by maximal allowed distance (1).

$$sim_{x,y,maxDist} = \max\left(0, \frac{maxDist - |x-y|}{maxDist}\right) \quad (1)$$

For *string* attributes (*movie name*) the similarity is defined as inverse of relative Levenshtein distance (2). This allows us to define as similar e.g. movie series.

$$sim_{x,y} = 1 - \left(\frac{levenshtein(x,y)}{\max\left(lenght(x),\ lenght(y)\right)}\right) \quad (2)$$

[1] www.omdbapi.com

Finally, similarity of set attributes (*genre*, *country*, *director* and *actors*) is defined as Jaccard similarity (3). Note that nominal attributes can be dealt as sets of size 1.

$$sim_{\mathbf{x},\mathbf{y}} = |(\mathbf{x} \cap \mathbf{y})| \,/\, |(\mathbf{x} \cup \mathbf{y})| \quad (3)$$

Differences between audiences of users are considered in the form of user bias (average value of engagement per user). The whole algorithm is presented in the pseudocode in Algorithm 1.

Algorithm 1: Hybrid biased k-NN algorithm: for tweet *tID*, its movie *mID* and fixed *k*, the algorithm first compute similarities to other movies and selects *k* most similar movies. Then for each tweet about the movie the predicted ranking $\hat{r}$ is increased according to similarity $\hat{s}$, user engagement *r* and bias of the tweeting user. The bias of the current movie is added in the final $\hat{r}$ prediction too.

```
function HybridBiasedKNN(tID, mID1, k){
 r̂ = 0;
 /*compute similarity for all movies */
 foreach(mID2 in TrainSet){
   S[mID2] = similarity(mID1, mID2);
 }
 S̄ = getKMostSimilar(S,k);
 /*get all tweets about movies in S̄ */
 foreach({uID, mID, r, ŝ}:
   {uID, mID, r} in TrainSet && S̄[mID]=ŝ){
    r̂ += ŝ * r / bias(uID);
 }
 r̂ = bias(mID1) + (r̂ / sum(ŝ))
 return r̂; }
```

Table 1: Results (nDCG@10) of the off-the-shelf algorithms.

Method	nDCG	Method	nDCG
Random predictions	*0.7482*	Item-Item k-NN	0.7604
Bi-Polar Slope One	0.7652	Decision Tree	0.7494
Factor Wise Matrix Factorization	0.7556	**Support Vector Machines (SVM)**	**0.8057**

3. EVALUATION

Some state-of-the-art prediction methods were used to serve as baseline (see Table 1). We used their implementation in RapidMiner Studio[2], or its Recommender extension [2]. Except for the SVM, those methods provided only minor improvements over the random predictions.

While evaluating Hybrid Biased k-NN we focused mainly on the utility of each attribute, using of user bias and also methods to combine results from multiple algorithm settings. Only a fraction of our results can be shown due to the space reasons. We can state that most of the attributes used as sole similarity measure provided good results, especially *IMDB metascore*, *director*, *language* and *country* (see Table 2). Also omitting user bias led to decrease of utility throughout various algorithm settings. The neighborhood size *k* between 50 and 100 provided good results. We also tried numerous variants of combining attribute similarities within the Hybrid k-NN algorithm (omitting some attributes, weighting schemas) and ensemble methods (stacking, linear regression, averaging), but so far the best results was achieved by average of Hybrid k-NN results based on single attribute, omitting single top and bottom result – see Table 3.

Table 2: Results (nDCG@10) of Hybrid biased k-NN algorithm using only single content-based attribute to compute similarity.

Avg rating	0.7918	Movie name	0.7947	Language	0.8005
Awards	0.7652	Date	0.7962	Director	0.8029
Metascore	**0.8057**	Genres	0.7919	Actors	0.7930
# of ratings	0.7964	Country	0.7984		

Table 3: Results (nDCG@10) of Hybrid biased k-NN algorithm. *No bias* stands for omitting user and item bias from the algorithm.

Method	nDCG
Hybrid k-nn (*Metascore*, *Language*, *Director*, *Country*, *Date*, *# of ratings*), k=65	0.7927
Hybrid k-nn(*Metascore*, *Language*, *Director*, *Country*, *Date*, *# of ratings*), k=65, no bias	0.7792
Linear Regression (*Metascore*, *Language*, *Director*, *Country*, *Date*, *# of ratings*)	0.7913
AVG (*Metascore*, *Language*, *Director*, *Country*, *Date*, *# of ratings***),** omit best and worst prediction	**0.8134**

4. CONCLUSIONS AND FUTURE WORK

In this paper, we proposed a variant of k-NN algorithm to predict movie tweets popularity (RecSys Challenge 2014 task). The algorithm outperformed examined state-of-the-art prediction techniques and resulted in the midfield of the challenge leaderboard. Some of our ideas didn't work as we expected, namely using more advanced ensemble techniques, using rank of tweets instead of their user engagement or omitting users with zero engagement. There are also several possible extensions to this work. So far we did not purse temporal dependence at all. Also some tweet characteristics or more movie content-based attributes can be employed as well.

The work on this paper was supported by the grant SVV-2014-260100, GAUK-126313 and P46. Hybrid k-NN source code is available on http://www.ksi.mff.cuni.cz/~peska/hybrid_knn.zip.

REFERENCES

[1] Forbes, P. & Zhu, M. Content-boosted matrix factorization for recommender systems: experiments with recipe recommendation. *In RecSys 2011, ACM*, **2011**, 261-264

[2] Mihelčić, M., Antulov-Fantulin, N., Bošnjak, M., Šmuc, T., Extending RapidMiner with recommender systems algorithms, *In RCM 2012*, Budapest, Hungary, **2012**

[3] Peska, L.; Vojtas, P.: Hybrid Recommending Exploiting Multiple DBPedia Language Editions, *In ESWC 2014 Linked Open Data-enabled Recommender Systems Challenge*, **2014**

[4] Peska, L. & Vojtás, P.: Recommending for Disloyal Customers with Low Consumption Rate. *In SOFSEM 2014, Springer, LNCS 8327*, **2014**, 455-465

[5] Ristoski, P.; Mencia, E.L. & Paulheim, H.: A Hybrid Multi-Strategy Recommender System Using Linked Open Data, *In ESWC 2014*, **2014**

[2] www.rapidminer.com